

LBIST_STEP_CLKC and LBIST_STEP_STEPE. The step clock signal LBIST_STEP_CLKC actually comprises three signals LBIST_STEP_CLKC1, LBIST_STEP_CLKC2, and LBIST_STEP_CLKC3. The LBIST_STEP_CLKC clock signal, normally high, enables the LBIST_STEP_CLKC1 through to the core latches (not shown) via the core logic clock splitters (not shown) of the core 900. The LBIST_STEP_CLKC2 is enabled by the LBIST_STEP_CLKC clock signal through the clock splitters (not shown) of the low power register array ("LPRA") wrappers 905. The LBIST_STEP_CLKC clock signal also enables the LBIST_STEP_CLKC3 through the clock splitter (not shown) of the wrappers for the memory components 150, *i.e.*, the SRAM wrappers 910.

Clock control is technically not a function within LBIST. Vendor ASICs have a primary input pin (not shown) on which they receive a TEST_MODE signal from the test controller 915 through the testing interface 180. When this signal is high, the LBIST is completely inhibited from affecting operation of vendor chip testers. During vendor chip LSSD testing, this input is held high. During normal operation, TEST_MODE is low. A signal received through the testing interface 180, *e.g.*, a LBIST_SEL signal from a joint test access group ("JTAG") controller 920, determines if the LBIST can supply the scan clock signals and step clock signals. The LBIST_SEL signal controls a multiplexer (not shown) between the system clock signal received through the testing interface 180 and the LBIST step clock signals. It also controls multiplexers (not shown) between the LSSD clock signals and the outputs of the clock splitters driven by the LBIST step clock signals as discussed above.

In the illustrated embodiment, the LBIST runtime is a function of the vector count provided the LBIST engine 110 and the hardwired scan length value discussed above. The number of clock cycles can be computed as:

$$([vector\ count \times (4 + (2 \times scan\ length\ value))] + 2)$$

The clock rate is determined by a clock signal provided through the testing interface 180, *e.g.*, the JTAG TCK.

Turning now to **FIG. 6**, the MBIST domain 170, first shown in **FIG. 1**, includes the MBIST engine 120 and a MBIST signature register 605 whose content is the MBIST signature 140. The MBIST engine 120, in the illustrated embodiment, comprises a series of alternative MBIST state machines 610—one of which drives a nested MBIST engine 620 in accordance with yet another aspect of the invention. In this particular embodiment, the nested MBIST engine 620 is provided by an ASIC vendor, and one of the MBIST state machines 610 is designed to operate with that particular vendor-supplied, nested MBIST engine 620. Indeed, each of the MBIST state machines 610 is designed to operate with one or more alternative vendor-supplied nested MBIST engines 620 that may be nested in the MBIST engine 120. The MBIST state machines 610 may also be modifiable to facilitate operation with vendor-supplied MBIST engines 620 that were not anticipated at the time the ASIC 150 was designed.

The MBIST engine 120 is therefore modifiable or configurable at the time the ASIC is implemented in a register transfer level (“RTL”) specification to accommodate a variety of nested MBIST engines 620 that might be obtained from various vendors. As those in the art having the benefit of this disclosure will appreciate, the nested MBIST engine 620 and the MBIST state machines 610 are a predefined library elements in standard RTL applications software. The RTL specification for the ASIC 100 contains a logic wrapper (not shown) defining the inputs and outputs for the library elements that define which of the MBIST state machines 610 provides the input and output to the nested MBIST engine 620. The RTL specification is then synthesized into a gate-level implementation for the ASIC 100.

The illustrated embodiment is therefore versatile with respect to which vendor-supplied MBIST engines 620 may be used. However, such versatility may not be desired in all embodiments. Some embodiments of the present invention may therefore include only a single MBIST state machine 610. Or, the versatility may be incorporated into a single MBIST state machine 610 that is highly modifiable or configurable. The number of MBIST state machines 610 employed in any given embodiment will therefore be implementation specific.

In accordance with still another aspect of the present invention, the results of the MBIST on the memory components 190 are stored as the MBIST signature 140, shown in

FIG. 1, within the MBIST signature register 605. The structure and function of the MBIST signature 140 are analogous to the structure and function of the LBIST signature 130. The MBIST signature register 140 is also a multiple input signature register, but its contents differ from the MISR 220, and so will be loaded differently. In this particular embodiment, paranoid checks and MBIST engine states are stored in the MBIST signature register 605 for debug purposes. One bit of the MBIST signature register 605, *e.g.*, the bit B_{31} , shown in **FIG. 7**, of this register is a “done” bit. The done bit indicates if the MBIST is done and, hence, if the results stored are new or resulted from a previous run.

The nested MBIST engine 620 tests from one to sixteen memory components 190 (not shown) in parallel depending on the specification of the ASIC vendor. The dual mode BIST controller 100 has a separate clock domain for the MBIST engine 120 in which the 150 MHz system clock signal is halved and the MBIST engine 120 is driven with the resultant 75 MHz clock signal. The results of the tests on the SRAMs are stored in the MBIST signature register 605. Bit B_{31} of this register is the “done” bit. The done bit indicates if the results stored are new or resulted from a previous run. In this particular embodiment, paranoid checks and MBIST engine states are also stored in the MBIST signature register 605 for debug purposes.

Each of the MBIST state machines 610 has, as is shown in **FIG. 8**, five states: a reset state 810, an initialization state 820, a flush state 830, a test state 840, and a done state 850. The MBIST engine 120 is reset to the reset state 810 by asserting the external reset signal. Note that, in this particular embodiment, the same external reset signal resets both the LBIST engine 110 and the MBIST engine 120.

The MBIST state machine 610 transitions to the initialization state 820 upon receipt of a MBIST select signal and a MBIST run signal received through the testing interface 180. The initialization state 820 is followed by a flush and then the test patterns as the MBIST engine 120 cycles through the initialization state 820, flush state 830, and test state 840. This transition occurs upon the completion of initialization of components and signals in the MBIST domain. The flush state 830 continues until the memory components 190 are flushed and initialized them to a known state. The MBIST state machine 610 then transitions to the test state 840. The MBIST engine 120 drives a one direction test pattern bus (not shown) out